Retour sur le TP 11 et des listes

Retour sur le TP 11

On considère une liste **A** dont les éléments sont des nombres entiers compris entiers compris entre 0 et 255.
On souhaite construire une liste **L** de taille 256 telle que, pour tout i tel que $0 \leqslant i \leqslant 255$, **L[i]** représente le nombre de fois où i apparait dans la liste **A**.

1. Proposer un exemple, de liste **A** et préciser alors quelle est la liste **L**.

2. Proposer un autre exemple.

3. Écrire une fonction **effectifs(A)** qui prend en paramètre la liste **A** et construit et renvoie la liste **L** associée.

4. Tester cette fonction avec les exemples proposés aux premières questions.

On cherche maintenant à séparer les effectifs en deux groupes d'effectifs comparables. On note $N =$ **len(A)**. On recherche donc l'indice j vérifiant :

$$L[0] + \cdots + L[i_0] \leqslant \frac{N}{2} \quad \text{et} \quad L[0] + \cdots + L[i_0] + L[i_0 + 1] > \frac{N}{2}$$

5. Reprendre les exemples des deux premières questions. Que vaut i_0 ?

6. Si vous n'êtes pas satisfaits par les exemples choisis auparavant, proposez-en un autre et donnez la liste **L** et la valeur de i_0.

7. Écrire une fonction **partage(L,N)** qui détermine et renvoie cet indice i_0.

Autres choses avec des listes

8. Écrire une fonction de la forme :

```
def somme(L):
    ...
    return s
```

qui calcule et renvoie la somme des termes d'une liste de nombres **L**.

9. Noter le résultat obtenu avec **print(somme(list(range(1,100))))**.

Réponse : _____

10. Tester le code suivant et comprendre à quoi correspond l'affichage obtenu :

```
L = [1,2,3,4,5,6,7]
T = []
k = 0
while k<len(L):
    T.append(L[k])
    k = k+1
print(T)
```

11. Modifier le code précédent pour que la liste **T** construite soit constituée en prenant un élément sur deux dans **L** (en commençant par l'élément d'indice 0). Vérifier que la liste **T** affichée est bien conforme.

Réponse : _____

12. Modifier le code précédent pour que la liste **T** construite soit constituée de tous les éléments de **L** mais apparaissant dans l'ordre inverse. Vérifier que la liste **T** affichée est bien conforme.

Réponse : _____

13. Écrire une fonction de la forme :

```
def retourner(L):
    ...
    return T
```

qui construit et renvoie la liste **T** constituée des éléments de **L** mais apparaissant dans l'ordre inverse.

14. Vérifier que **print(retourner([1,2,3]))** et **print(retourner([]))** affichent des résultats conformes.

Réponse : _____

15. Écrire une fonction de la forme :

```
def eliminer_0_20(L):
    ...
    return T
```

qui construit et renvoie la liste **L** constituée uniquement des éléments de **L** qui sont compris, au sens large, entre 0 et 20.

16. Vérifier que les résultats affichés avec

```
print(eliminer_0_20([]))
print(eliminer_0_20([2,-1,10,15,25]))
```

sont bien conformes.

Réponse : _____

17. Tester le code suivant et comprendre à quoi correspond l'affichage obtenu :

```
L = [1,2,-3,6,-7]
for i in range(0,len(L)):
    if L[i]<0:
        L[i] = -L[i]
print(L)
```

18. Écrire une fonction de la forme :

```
def recadrer_0_20(L):
    ...
```

qui modifie la liste **L** de la manière suivante : les éléments de **L** compris entre 0 et 20 au sens large sont inchangés, les éléments de **L** strictement négatifs sont remplacés par 0 et les éléments de **L** strictement supérieurs à 20 sont remplacés par 20.
Cette fonction ne renverra pas de résultat, elle modifie directement la liste **L** passée en paramètre.

19. Vérifier ensuite que le code :

```
L = [2,-1,10,15,25]
recadrer_0_20(L)
print(L)
```

affiche un résultat conforme.

Réponse : _____

Corrections

Q 3.

```
def effectifs(A):
    L = [0]*256
    for j in range(0,len(A)):
        L[A[j]] = L[A[j]]+1
    return L
```

Q 7.

```
def partage(L,N):
    i0 = 0
    s = L[0]
    while s<=N/2:
        i0 = i0+1
        s = s+L[i0]
    return i0-1
```

Q 8.

```
def somme(L):
    s = 0
    for k in range(0,len(L)):
        s = s+L[k]
    return s
```

Q 9.

```
print(somme(list(range(1,100))))
```

```
4950
```

Q 11.

```
L = [1,2,3,4,5,6,7]
T = []
k = 0
while k<len(L):
    T.append(L[k])
    k = k+2
print(T)
```

```
[1, 3, 5, 7]
```

Q 12.

```
L = [1,2,3,4,5,6,7]
T = []
k = 0
while k<len(L):
    T.append(L[len(L)-k-1])
    k = k+1
print(T)
```

```
[7, 6, 5, 4, 3, 2, 1]
```

Q 13.

```
def retourner(L):
    T = []
    for k in range(0,len(L)):
        T.append(L[len(L)-k-1])
    return T
```

Q 14.

```
print(retourner([1,2,3]))
print(retourner([]))
```

```
[3, 2, 1] []
```

Q 15.

```
def eliminer_0_20(L):
    T = []
    for k in range(0,len(L)):
        if 0<=L[k]<=20:
            T.append(L[k])
    return T
```

Q 16.

```
print(eliminer_0_20([]))
print(eliminer_0_20([2,-1,10,15,25]))
```

```
[] [2, 10, 15]
```

Q 18.

```
def recadrer_0_20(L):
    for k in range(0,len(L)):
        if L[k]<0:
            L[k] = 0
        elif L[k]>20:
            L[k] = 20
```

Q 19.

```
L = [2,-1,10,15,25]
recadrer_0_20(L)
print(L)
```

```
[2, 0, 10, 15, 20]
```